Click/report event from FLEDGE component ads

shivanisha@chromium.org
Status: Posted on github
Visibility: public
Aug-Sep 2022

References

Turtledove github issues: https://github.com/WICG/turtledove/issues/332, https://github.com/WICG/turtledove/issues/281

Issue

The issue here points out a reasonable use case where the user clicks on a component ad and the ad-tech server would want to be notified of that via fenced frames ads reporting mechanism. Note that this mechanism is by design built to support event-level reporting and a click on a nested component ad is a significant enough event that would need to be supported.
However reportEvent() only works if there is an associated registerAdBeacon() call and that metadata is only associated with the root fenced frame (the parent ad). If we were to also associate that metadata with the nested ads then that would imply that the server can join arbitrary data from the component ads and the parent ad, which is not a desired FLEDGE information flow (more details below). Note that reportEvent() can be called anytime and with arbitrary data and is not gated on a user click/gesture.

Related issue : event level attribution reporting

A related issue is how a click in the component ad can be used for attribution reporting.
To allow event level attribution reporting, the mechanism used by ad-tech is that reportEvent() from the parent FF carries a generated id which is also associated with the click based navigation and that helps join the 2 at the server side and attribute the navigation to the specific ad slot (fenced frame).
This can be fixed if reportEvent() were allowed from the component fenced frame. It can then create its own id and that can also be joined at the server side to the already joined <id-in-registerAdBeacon, id-in-parent-fenced-frame> since this reportEvent will be resulting in a ping to the same reporting url as the parent fenced frame.

Privacy impact of reportEvent from component ad

The fenced frames ads reporting (FFAR) mechanism is only allowed from reportWin or reportResult, neither of which have the component ads information, so the combined event level reporting from reportWin, reportResult and FFAR does not allow a join of the main ad and components ads.
If reportEvent() was allowed from a component ad, the combination of main ad and component ads would now be known at the server which goes against the privacy assumption where each

of the main and component urls are k-anonymous and their join would not necessarily be k-anonymous.

Proposed fix and privacy impact

A potential approach could be to allow reportEvent from the component fenced frame but only if it received a click/transient user activation. To further reduce the amount of information leaked we will restrict it to at most one such reportEvent among all the component ads fenced frames for a given parent fenced frame.

Privacy-wise this can lead to at most 2 k-anonymous urls to be joined (that of the parent and the component that was clicked) but that seems acceptable as this additional information is gated on user activation on that component ad.

Utility wise it will fix the attribution reporting as well as click reporting issues mentioned above and the max of one event should be sufficient to serve most use cases since a click will likely lead to a navigation.

Alternatives considered

An alternative could be to propagate the click event from component ad fenced frames to the parent fenced frame. Since it will be gated on user activation and does not carry arbitrary information, it is ok to have this limited communication channel between the nested component FF and its parent. However see below for concerns with this approach:

Concerns:

- On the web platform click events don't get bubbled across the fenced frame boundary as well as across cross-origin iframe boundary while such an approach will require to diverge from the web platform norm around cross-origin iframes.
- The other concern in this approach is possible complexity around the ordering of the parent fenced frame receiving the click event and calling reportEvent while the click might lead to the navigation of the top-level page.

Because of the concerns above, instead of bubbling a click event, another approach could be adding an API which only works for component ads, as below:

window.fence.reportClick()

This API when invoked from the component ad fenced frame will let the browser know to invoke the same functionality as if window.fence.reportEvent('click') was invoked from the parent frame. The important difference between this and a reportEvent call is that there is no arbitrary data and arbitrary values of event type allowed. This however does not fix the attribution reporting issue mentioned above because there is no associated id in the reportEvent ping and the navigation resulting from the click.